Mon. Not. R. Astron. Soc. **000**, 1–9 (2015) Printed 18 March 2015 (MN LATEX style file v2.2)

Distributed Radio Interferometric Calibration

Sarod Yatawatta
ASTRON, Postbus 2, 7990 AA Dwingeloo, the Netherlands

18 March 2015

ABSTRACT
Increasing data volumes delivered by a new generation of radio interferometers require computationally efficient and robust calibration algorithms. In this paper, we propose distributed calibration as a way of improving both computational cost as well as robustness in calibration. We exploit the data parallelism across frequency that is inherent in radio astronomical observations that are recorded as multiple channels at different frequencies. Moreover, we also exploit the smoothness of the variation of calibration parameters across frequency. Data parallelism enables us to distribute the computing load across a network of compute agents. Smoothness in frequency enables us reformulate calibration as a consensus optimization problem. With this formulation, we enable flow of information between compute agents calibrating data at different frequencies, without actually passing the data, and thereby improving robustness. We present simulation results to show the feasibility as well as the advantages of distributed calibration as opposed to conventional calibration.

Key words: Instrumentation: interferometers; Methods: numerical; Techniques: interferometric

1 INTRODUCTION

Many of the science drivers in modern radio astronomy seek weak signals buried in noise and bright foregrounds. Existing radio interferometers are upgraded and new ones are being built to deliver large volumes of data to achieve this goal. A major step of data processing in such telescopes is the correction of systematic errors and the removal of contaminating foregrounds from the data, which is called *calibration*. With wide fields of view, calibration has to be done along hundreds of directions in the sky, especially at low radio frequencies (Bregman 2012). This entails solving for a large number of unknowns and a reliable solution can only be obtained if there are sufficient constraints.

There are several novel calibration algorithms (Kazemi et al. 2011; Kazemi & Yatawatta 2013; Yatawatta 2013; Tasse 2014) that are presently being used that improve speed and robustness in calibration. Most of these algorithms (Kazemi et al. 2011; Kazemi & Yatawatta 2013; Yatawatta 2013) use an algebraic data model that directly solve for Jones matrices representing the cumulative effect of the systematic errors. On the other hand, solving for a physical model (Bregman 2012; Tasse 2014) would reduce the number of unknowns, especially since most of the systematic errors are known to have a smooth variation across frequency. One drawback of the physical model based calibration is the need to access data across a wide frequency range, which is computationally not feasible at a central location given that there are thousands of frequency channels in the data. Moreover, a physical model requires an accurate description of the frequency dependence and this can only be done for specific and well studied errors. Therefore, in this paper we propose a distributed calibration scheme that preserve the simplicity and computational speed of algebraic model based calibration while enforcing the smoothness of the calibration parameters across frequency. This can be thought of as getting the best of both aforementioned calibration approaches. In order to do this, we reformulate calibration as a distributed optimization problem and use *consensus optimization* (Boyd et al. 2011).

Distributed learning and distributed optimization (Tsitsiklis 1984; Bertsekas & Tsitsiklis 1997) is a widely researched topic in various disciplines. Consensus optimization (Boyd et al. 2011) is one algorithm for distributed learning. In the era of exascale computing and big data, the importance of such algorithms grow ever more (for some recent results see for instance Chang et al. (2014); Wei & Ozdaglar (2012); Mota et al. (2013)). Instead of one compute agent accessing data across all frequencies (which is computationally unfeasible), we consider a situation where a group of compute agents accessing data across smaller frequency intervals. This matches ideally with how radio astronomical data is organized (data for the full observing bandwidth is divided into channels and channels are grouped into subbands), and stored. Therefore, we consider a situation where each compute agent having access to only a few subbands (while the full bandwidth consists of a few hundred subbands). Each compute agent will calibrate the data available

locally (using an algebraic model) and the calibration solutions are transferred to a centralized location (fusion center). At the fusion center, consensus on the smoothness of the parameters across frequency is enforced. Afterwards, this update is passed back to each compute agent. Therefore, indirectly, each compute agent receives information across the whole frequency range, thus improving the calibration. Moreover, since no attempt is made to directly model or estimate underlying physical parameters, the calibration algorithms are simpler and less susceptible to model errors. Furthermore, the amount of information that needs to be exchanged between the fusion center and the compute agents is much less compared to the amount of data being calibrated, making this scheme computationally feasible. We also note that similar approaches have been proposed and tested for radio astronomical image synthesis (Ferrari et al. 2014; Carrillo et al. 2014) to reduce the number of Fourier space samples used in imaging as well as to improve the quality of reconstruction. Such imaging approaches would certainly complement the calibration approach proposed in this paper.

The rest of the paper is organized as follows: In section 2, we introduce radio interferometric calibration and in section 3, we reformulate it as a distributed consensus optimization problem. We give results based on simulations in section 4 to show the feasibility and superiority of the proposed scheme and draw our conclusions in section 5.

Notation: Lower case bold letters refer to column vectors (e.g. $\boldsymbol{y}$). Upper case bold letters refer to matrices (e.g. C). Unless otherwise stated, all parameters are complex numbers. The set of complex numbers is given as $\mathbb{C}$ while the set of real numbers as $\mathbb{R}$. The matrix pseudo-inverse, transpose, and Hermitian transpose are referred to as $(.)^\dagger$, $(.)^T$, $(.)^H$, respectively. The matrix Kronecker product is given by $\otimes$. The identity matrix is given by I. Estimated parameters are denoted by a hat, $\widehat{(.)}$. The Frobenius norm is given by $\|.\|$. A uniform distribution in $[0,1]$ is given as $\mathcal{U}(0,1)$.

2 RADIO INTERFEROMETRIC CALIBRATION

In this section we give a brief overview of the data model used in radio interferometric calibration (Hamaker et al. 1996; Thompson et al. 2001). We consider the radio frequency sky that is part of the sky model to be composed of discrete sources, far away from the earth such that the approaching radiation from each one of them appears to be plane waves. However, in reality there is large scale diffuse structure as well. There are N receiving elements with dual polarized feeds in the array and at the p-th station, this plane wave causes an induced voltage, which is dependent on the beam attenuation as well as the radio frequency receiver chain attenuation. Consider the correlation of signals at the p-th receiver and the q-th receiver, with proper signal delay at frequency f and time t (with finite bandwidth and integration time). After correlation, the correlated signal of the p-th station and the q-th station (named as the *visibilities*), $\mathsf{V}(p,q,t,f)$ ($\in \mathbb{C}^{2\times 2}$) is given by

$$\mathsf{V}(p,q,t,f) = \sum_{k=1}^{K} \mathsf{J}(p,k,t,f)\mathsf{C}(p,q,k,t,f)\mathsf{J}(q,k,t,f)^H + \mathsf{N}_{pq}. \tag{1}$$

In (1), $\mathsf{J}(p,k,t,f)$ and $\mathsf{J}(q,k,t,f)$ are the Jones matrices describing errors along the direction of source k, at stations p and q, at time t and frequency f, respectively. These matrices represent the effects of the propagation medium, the beam shape and the receiver. There are K sources in the sky model and the noise matrix is given as N_{pq} ($\in \mathbb{C}^{2\times 2}$). The contribution from the k-th source on baseline pq is given by the coherency matrix $\mathsf{C}(p,q,k,t,f)$ ($\in \mathbb{C}^{2\times 2}$). We consider the sources in the sky model to be unpolarized and for the k-th direction, with intensity $I(p,q,k,f)$ (invariant over time but dependent on p,q if the source is resolved) we have

$$\mathsf{C}(p,q,k,t,f) = e^{\jmath\phi(p,q,k,t,f)} \begin{bmatrix} I(p,q,k,f) & 0 \\ 0 & I(p,q,k,f) \end{bmatrix} \tag{2}$$

where $\phi(p,q,k,t,f)$ is the Fourier phase component that depends on the direction in the sky as well as the separation of stations p and q and can be exactly calculated. Moreover, it is also possible to refine $\mathsf{C}(p,q,k,t,f)$ to include finite integration time and bandwidth (Thompson et al. 2001) but in this paper we use the simpler form. The noise matrix N_{pq} is normally assumed to have elements with zero mean, complex Gaussian entries with equal variance in real and imaginary parts but the statistics will vary because of the unmodeled structure (Kazemi & Yatawatta 2013).

Calibration is the estimation of a set of parameters $\boldsymbol{\theta}$ that describe the Jones matrices $\mathsf{J}(p,k,t,f)$ for $p \in [1,N]$ and $k \in [1,K]$ for given t and f. The solutions obtained are additionally used to correct the data and also to calculate the residual by subtracting the predicted model from the (corrected) data. The maximum likelihood (ML) estimate of $\boldsymbol{\theta}$ under zero mean, white Gaussian noise is obtained by minimizing the least squares cost function

$$g(\boldsymbol{\theta}) = \sum_{t,f}\sum_{p,q} \| \mathsf{V}(p,q,t,f) - \sum_{k=1}^{K} \mathsf{J}(p,k,t,f)\mathsf{C}(p,q,k,t,f)\mathsf{J}(q,k,t,f)^H \|^2 \tag{3}$$

and can be improved by using a weighted least squares estimator to account for errors in the sky model (Kazemi & Yatawatta 2013). At this point, we make several points clear and make certain assumptions:

- The solutions $\boldsymbol{\theta}$ are assumed to be invariant over time, within the time interval $g(\boldsymbol{\theta})$ is minimized, therefore from now on, we drop the time dependence from $\mathsf{J}(p,k,t,f)$ and use $\mathsf{J}(p,k,f)$ instead. This can also be done for $\mathsf{C}(p,q,k,t,f)$ to have $\mathsf{C}(p,q,k,f)$ and $\mathsf{V}(p,q,t,f)$ to have $\mathsf{V}(p,q,f)$, because the geometry of baseline pq is dependent on t (also the summation over t is implicitly assumed but not explicitly stated).
- The solutions for different directions k are assumed to have statistically independent noise, therefore, we use expectation maximization (EM) and space alternating expectation maximization (SAGE) (Fessler & Hero 1994; Kazemi et al. 2011) to simplify the cost function in (3) over summation in k.

• We *do* assume variability of the solutions over f, indeed, this is the novelty of this paper. For some directions, we can assume a smooth variation of the underlying parameters over f as done in (Tasse 2014). However, the drawback of the approach taken in (Tasse 2014) is the amount of data needed to get a reliable estimate of this variation over f. Indeed for a telescope like LOFAR that observe over a wide bandwidth, producing hundreds of subbands and thousands of channels of data, even storing all subbands at one location is problematic, let alone reading that data into memory.

Therefore, in this paper, we reformulate calibration as a distributed optimization problem. We assume smooth variation of the parameters over f, but unlike in (Tasse 2014), we do not directly estimate those underlying parameters. In other words, the smooth variation is imposed as an additional constraint, but the calibration problem is still kept unchanged by estimating $\mathsf{J}(p,k,f)$ for each p, k and f. Note that since the storage of data is by default distributed over f, i.e. different subbands (channels) are stored at different locations, the optimization can also be done in a distributed way. This distribution of computations does not necessarily reduce the total computational cost, but it can reduce the computational cost required at any one location where data is stored, provided that the computations only access the data that is locally available. In the following section, we describe how this can be done.

3 DISTRIBUTED CALIBRATION

Consider the Jones matrices along the k-th direction, $\mathsf{J}(p,k,f)$, for N stations, let

$$\mathsf{J}_{kf} \triangleq \begin{bmatrix} \mathsf{J}(1,k,f) \\ \mathsf{J}(2,k,f) \\ \vdots \\ \mathsf{J}(N,k,f) \end{bmatrix} \tag{4}$$

where J_{kf} ($\in \mathbb{C}^{2N\times 2}$) is the augmented Jones matrix. Also define a canonical selection matrix A_p ($\in \mathbb{C}^{2\times 2N}$)

$$\mathsf{A}_p \triangleq [\mathbf{0},\mathbf{0},\ldots,\mathsf{I},\ldots,\mathbf{0}]. \tag{5}$$

where all elements of A_p are zero except the p-th block which is an identity matrix. Using A_p and J_{kf}, we can recover $\mathsf{J}(p,k,f) = \mathsf{A}_p\mathsf{J}_{kf}$.

The ML estimate for $\boldsymbol{\theta}$ can ideally be obtained by minimizing (3), but this needs access to all data. In *normal* calibration, solutions are obtained separately for each f, using data at that frequency. For given f, consider partitioning the parameters as $\{\boldsymbol{\theta}_{kf} : k = 1\ldots K\}$. We apply the EM/SAGE algorithm (Kazemi et al. 2011) to estimate each $\boldsymbol{\theta}_{kf}$. The *expectation* step in SAGE finds the visibility contribution V_{pqkf} from $\mathsf{V}(p,q,f)$ (with $\mathsf{C}_{pqkf} = \mathsf{C}(p,q,k,f)$) as

$$\mathsf{V}_{pqkf} = \mathsf{V}(p,q,f) - \sum_{l,l\neq k} \mathsf{A}_p\mathsf{J}_{lf}\mathsf{C}_{pqlf}\left(\mathsf{A}_q\mathsf{J}_{lf}\right)^H \tag{6}$$

and using this, in the *maximization* step, the current estimate for $\boldsymbol{\theta}_{kf}$ is obtained by minimizing

$$g_{kf}(\boldsymbol{\theta}_{kf}) = \sum_{p,q} \|\mathsf{V}_{pqkf} - \mathsf{A}_p\mathsf{J}_{kf}\mathsf{C}_{pqkf}\left(\mathsf{A}_q\mathsf{J}_{kf}\right)^H\|^2. \tag{7}$$

Now, to simplify the description even further, we only consider calibration along the k-th direction or minimizing $g_{kf}(\boldsymbol{\theta}_{kf})$, so we drop the subscript k. Let $\boldsymbol{\theta}_{kf} = \mathsf{J}_{kf} = \mathsf{J}_f$ where J_f is defined as in (4). Thereafter, we have the simplified form for (7)

$$g_f(\mathsf{J}_f) = \sum_{p,q} \|\mathsf{V}_{pqf} - \mathsf{A}_p\mathsf{J}_f\mathsf{C}_{pqf}\left(\mathsf{A}_q\mathsf{J}_f\right)^H\|^2. \tag{8}$$

So far, we have not imposed the smoothness over f to J_f, in order to do that, we introduce hidden variables Z_l ($\in \mathbb{C}^{2N\times 2}$), $l \in [1,F]$, and we enforce the relationship

$$\mathsf{J}_f = \sum_l b_l(f)\mathsf{Z}_l \tag{9}$$

onto J_f. In (9), the only frequency dependence on the right hand side is introduced by real scalar values $b_l(f)$ that can be thought of as polynomial terms (in f). The order of the polynomial is $F-1$ (where $F > 1$) and this controls the smoothness. For instance, given reference frequency f_0, we can select $b_l(f) = \left(\frac{f-f_0}{f_0}\right)^{l-1}$, but this is one possible polynomial and we can use more sophisticated expressions if needed.

If $\boldsymbol{b}_f$ ($\in \mathbb{R}^{F\times 1}$) is a vector representing all polynomial terms

$$\boldsymbol{b}_f = [b_1(f)\ b_2(f)\ \ldots\ b_F(f)]^T \tag{10}$$

we can rewrite (9) as

$$\mathsf{J}_f = \left(\boldsymbol{b}_f^T \otimes \mathsf{I}_{2N}\right)\mathsf{Z} = \mathsf{B}_f\mathsf{Z} \tag{11}$$

where I_{2N} is the $2N\times 2N$ identity matrix, $\mathsf{B}_f = \left(\boldsymbol{b}_f^T \otimes \mathsf{I}_{2N}\right)$ ($\in \mathbb{R}^{2N\times 2FN}$) and Z ($\in \mathbb{C}^{2FN\times 2}$) is the augmented matrix of hidden variables

$$\mathsf{Z} = [\mathsf{Z}_1^T\ \mathsf{Z}_2^T \ldots \mathsf{Z}_F^T]^T. \tag{12}$$

For each direction k, by imposing smoothness, we can find a set of hidden variables Z for any given value for F. At this point, we distinguish between direct estimation of Z and the method proposed in this paper:

• Centralized calibration is estimating Z directly from the data. However, this requires access to all frequencies (or at least a set of frequencies more than F) as shown in Fig. 1 (a). More rigorously, centralized calibration is estimating Z such that $\sum_f g_f(\mathsf{J}_f)$ is minimized. As we explained before, this is computationally not feasible because of the large data volumes needed.

• Instead of centralized calibration, we formulate distributed calibration as follows. Let there be P computational agents (or nodes) in a network. We assume the i-th agent will only have access to the data at frequency f_i as in Fig. 1 (b). However, we enforce *consensus* among all agents, in other words, we enforce an additional constraint $\mathsf{J}_{f_i} = \mathsf{B}_{f_i}\mathsf{Z}$ that all agents have to satisfy. Note that the total number of frequencies that the data is taken will almost surely be higher than P. In that case, we consider calibration of a subset of P frequencies selected from the total available frequencies. This selection has to be repeated sequentially until all the frequencies are calibrated.

With this network setup, we formulate distributed cal-

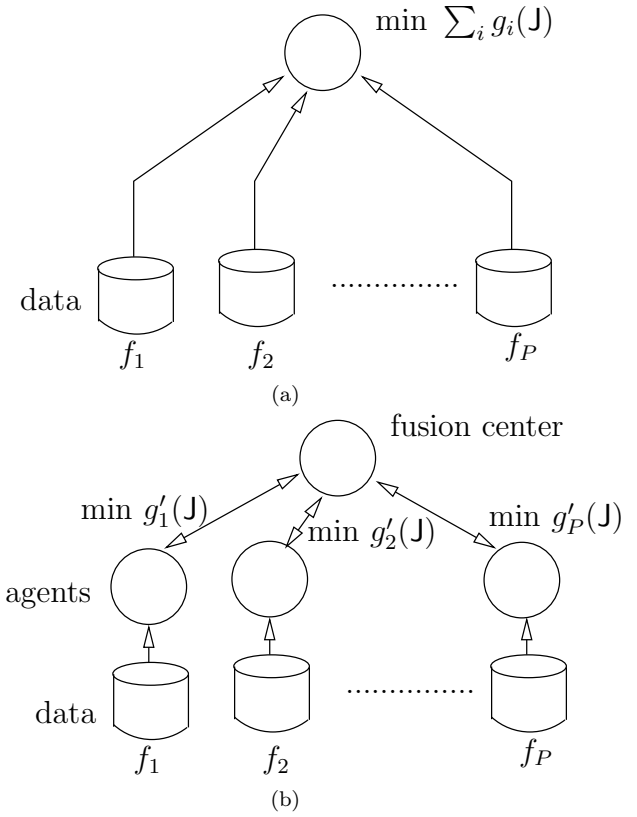


Figure 1. Centralized calibration compared with distributed calibration. (a) Centralized calibration requires access to data observed at multiple frequencies. (b) Distributed calibration uses agents that operate on data taken at only a single frequency but via a fusion center, information is passed to other agents operating on data at different frequencies. The exact functions minimized in centralized calibration $g_i(\mathsf{J})$ and distributed calibration $g_i'(\mathsf{J})$ are slightly different. In normal calibration, each agent in (b) operate independently without communicating with the fusion center or any other agent.

ibration as

$$\{\mathsf{J}_{f_1}, \mathsf{J}_{f_2}, \ldots, \mathsf{Z}\} = \arg\min_{\mathsf{J}_{f_i},\ldots,\mathsf{Z}} \sum_i g_{f_i}(\mathsf{J}_{f_i}) \quad \text{subject to } \mathsf{J}_{f_i} = \mathsf{B}_{f_i}\mathsf{Z}, \quad i \in [1, P] \tag{13}$$

which is actually a consensus optimization problem (Boyd et al. 2011). To solve this, we use the augmented Lagrangian method with the Lagrangian

$$\begin{aligned} &L(\mathsf{J}_{f_1}, \mathsf{J}_{f_2}, \ldots, \mathsf{Z}, \mathsf{Y}_{f_1}, \mathsf{Y}_{f_2}, \ldots) \\ &= \sum_i g_{f_i}(\mathsf{J}_{f_i}) + \|\mathsf{Y}_{f_i}^H(\mathsf{J}_{f_i} - \mathsf{B}_{f_i}\mathsf{Z})\| + \frac{\rho}{2}\|\mathsf{J}_{f_i} - \mathsf{B}_{f_i}\mathsf{Z}\|^2 \\ &= \sum_i L_i\left(\mathsf{J}_{f_i}, \mathsf{Z}, \mathsf{Y}_{f_i}\right) \end{aligned} \tag{14}$$

where Y_{f_i} are the Lagrange multipliers and ρ is the regularization factor. In order to solve (14), we use the consensus alternating direction method of multipliers (C-ADMM) (Boyd et al. 2011). If superscript n denote values at the n-th C-ADMM iteration, the values for the $(n+1)$-th iteration are updated as

$$(\mathsf{J}_{f_i})^{n+1} = \min_{\mathsf{J}} L_i\left(\mathsf{J}, (\mathsf{Z})^n, (\mathsf{Y}_{f_i})^n\right) \tag{15}$$

$$(\mathsf{Z})^{n+1} = \min_{\mathsf{Z}} \sum_i L_i\left((\mathsf{J}_{f_i})^{n+1}, \mathsf{Z}, (\mathsf{Y}_{f_i})^n\right) \tag{16}$$

$$(\mathsf{Y}_{f_i})^{n+1} = (\mathsf{Y}_{f_i})^n + \rho((\mathsf{J}_{f_i})^{n+1} - \mathsf{B}_{f_i}(\mathsf{Z})^{n+1}). \tag{17}$$

The minimization (15) has no closed form solution and needs to be done iteratively, for instance by using the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm (Nocedal & Wright 1999) or by using trust-region algorithms. In this paper, we use the Riemannian trust-region algorithm (RTR) described in Absil et al. (2007) for this minimization, and we need to calculate the gradient and the Hessian. The gradient and the Hessian with respect to J_{f_i} of (15) are given as (see Yatawatta (2013) and appendix A for proof)

$$\mathrm{grad}_i(L_i, \mathsf{J}) = \mathrm{grad}_i\left(g_{f_i}(\mathsf{J}), \mathsf{J}\right) + \frac{1}{2}\mathsf{Y}_{f_i} + \frac{\rho}{2}\left(\mathsf{J} - \mathsf{B}_{f_i}\mathsf{Z}\right) \tag{18}$$

and

$$\mathrm{Hess}_i(L_i, \mathsf{J}, \boldsymbol{\eta}) = \mathrm{Hess}_i\left(g_{f_i}(\mathsf{J}), \mathsf{J}, \boldsymbol{\eta}\right) + 0 + \frac{\rho}{2}\boldsymbol{\eta} \tag{19}$$

where we use (8) to get

$$\begin{aligned} &\mathrm{grad}_i\left(g_{f_i}(\mathsf{J}), \mathsf{J}\right) \\ &= -\sum_{p,q}\Big(\mathsf{A}_p^T(\mathsf{V}_{pqf_i} - \mathsf{A}_p\mathsf{J}\mathsf{C}_{pqf_i}\mathsf{J}^H\mathsf{A}_q^T)\mathsf{A}_q\mathsf{J}\mathsf{C}_{pqf_i}^H \\ &\quad + \mathsf{A}_q^T(\mathsf{V}_{pqf_i} - \mathsf{A}_p\mathsf{J}\mathsf{C}_{pqf_i}\mathsf{J}^H\mathsf{A}_q^T)^H\mathsf{A}_p\mathsf{J}\mathsf{C}_{pqf_i}\Big) \end{aligned} \tag{20}$$

and

$$\begin{aligned} &\mathrm{Hess}_i\left(g_{f_i}(\mathsf{J}), \mathsf{J}, \boldsymbol{\eta}\right) \\ &= \sum_{p,q}\Big(\mathsf{A}_p^T\Big((\mathsf{V}_{pqf_i} - \mathsf{A}_p\mathsf{J}\mathsf{C}_{pqf_i}\mathsf{J}^H\mathsf{A}_q^T)\mathsf{A}_q\boldsymbol{\eta} \\ &\quad -\mathsf{A}_p(\mathsf{J}\mathsf{C}_{pqf_i}\boldsymbol{\eta}^H + \boldsymbol{\eta}\mathsf{C}_{pqf_i}\mathsf{J}^H)\mathsf{A}_q^T\mathsf{A}_q\mathsf{J}\Big)\mathsf{C}_{pqf_i}^H \\ &\quad +\mathsf{A}_q^T\Big((\mathsf{V}_{pqf_i} - \mathsf{A}_p\mathsf{J}\mathsf{C}_{pqf_i}\mathsf{J}^H\mathsf{A}_q^T)^H\mathsf{A}_p\boldsymbol{\eta} \\ &\quad -\mathsf{A}_q(\mathsf{J}\mathsf{C}_{pqf_i}\boldsymbol{\eta}^H + \boldsymbol{\eta}\mathsf{C}_{pqf_i}\mathsf{J}^H)^H\mathsf{A}_p^T\mathsf{A}_p\mathsf{J}\Big)\mathsf{C}_{pqf_i}\Big). \end{aligned} \tag{21}$$

Minimization of (16) can be done in closed form. We take the derivative to get

$$\mathrm{grad}(L, \mathsf{Z}) = \sum_i \mathsf{B}_{f_i}^T\left(-\mathsf{Y}_{f_i} + \rho(-\mathsf{J}_{f_i} + \mathsf{B}_{f_i}\mathsf{Z})\right) \tag{22}$$

and equating this to zero gives us

$$\mathsf{Z} = \left(\sum_i \rho\mathsf{B}_{f_i}^T\mathsf{B}_{f_i}\right)^\dagger\left(\sum_i \mathsf{B}_{f_i}^T(\mathsf{Y}_{f_i} + \rho\mathsf{J}_{f_i})\right) \tag{23}$$

which can be further simplified by using $\mathsf{B}_{f_i} = \left(\boldsymbol{b}_{f_i}^T \otimes \mathsf{I}_{2N}\right)$ to get

$$\mathsf{Z} = \frac{1}{\rho}\left(\left(\sum_i \boldsymbol{b}_{f_i}\boldsymbol{b}_{f_i}^T\right)^\dagger \otimes \mathsf{I}_{2N}\right)\left(\sum_i \boldsymbol{b}_{f_i} \otimes (\mathsf{Y}_{f_i} + \rho\mathsf{J}_{f_i})\right). \tag{24}$$

Each column of Z in (24) can be written as $\boldsymbol{z} = (\mathsf{P} \otimes \mathsf{I}_{2N})\,\boldsymbol{r}$, where $\mathsf{P} = \frac{1}{\rho}\left(\sum_i \boldsymbol{b}_{f_i}\boldsymbol{b}_{f_i}^T\right)^\dagger$ $(\in \mathbb{R}^{F\times F})$ and $\boldsymbol{r} \in \mathbb{C}^{2FN\times 1}$ is the corresponding column of the right hand sum of (24). We can reshape $\boldsymbol{r}$ to get $\widetilde{\mathsf{R}} \in \mathbb{C}^{2N\times F}$. Then we can rewrite

$\boldsymbol{z} = \mathrm{vec}\left(\mathsf{I}_{2N}\widetilde{\mathsf{R}}\mathsf{P}^T\right) = \mathrm{vec}\left(\widetilde{\mathsf{R}}\mathsf{P}^T\right)$ which is far simpler to obtain than directly solving (24). Moreover, we see that from (24) in order to have full rank, the summation $\sum_i \boldsymbol{b}_{f_i}\boldsymbol{b}_{f_i}^T$ should at least have F terms (because the size of $\boldsymbol{b}_{f_i}\boldsymbol{b}_{f_i}^T$ is $F \times F$). In other words, we need to have data for at least F different frequencies, or $P \geqslant F$.

To recapitulate, we consider P compute agents operating simultaneously. Each agent i only has access to the data at frequency f_i. There is also a data fusion center with which each agent does communication. Consider calibration along a single direction first. Each agent i needs to estimate J_{f_i} ($\in \mathbb{C}^{2N\times 2}$) and will keep auxiliary variable Y_{f_i} ($\in \mathbb{C}^{2N\times 2}$) locally. The fusion center will keep the global variable Z ($\in \mathbb{C}^{2FN\times 2}$) and will pass the product $\mathsf{B}_{f_i}\mathsf{Z}$ ($\in \mathbb{C}^{2N\times 2}$) onto the i-th agent. With this additional variables, the C-ADMM algorithm for a single direction ($K = 1$) can be described as:

(S1) Each agent i finds estimate for J_{f_i} by solving (15). Thereafter, it sends back the result $\mathsf{Y}_{f_i} + \rho\mathsf{J}_{f_i}$ to the fusion center.

(S2) At the fusion center, after collecting the values $\mathsf{Y}_{f_i} + \rho\mathsf{J}_{f_i}$ from all agents, (16) is minimized by solving (24). Once the updated Z is obtained, $\mathsf{B}_{f_i}\mathsf{Z}$ is sent back to the i-th agent.

(S3) At agent i, Y_{f_i} is updated by using (17) with the new value of $\mathsf{B}_{f_i}\mathsf{Z}$ received from the fusion center. If stopping criteria (such as the maximum C-ADMM iterations) are not met, we go back to (S1) above.

Note that steps (S1) and (S3) above are done simultaneously at each agent. The centralized step (S2) is only an averaging step which is far less expensive compared with the minimization in (S1).

The above description is only for calibration along a single direction. In order to apply the same method for calibration along K directions, we only need slight modifications to the steps described above. We use subscript k to indicate the k-th direction. Each agent i needs to estimate K values J_{kf_i}. Moreover, each agent has K auxiliary variables Y_{kf_i}. The fusion center keeps the global variables Z ($\in \mathbb{C}^{2KFN\times 2}$) which has K blocks (let us denote the k-th block of Z as $(\mathsf{Z})_k$), one for each direction. Therefore, we have the C-ADMM algorithm for K directions as:

(D1) Each agent i finds estimate for K values J_{kf_i}. This is done by decomposing the K direction problem onto K problems of the type (15). In order to do this, we use SAGE algorithm (Kazemi et al. 2011). Note that in SAGE algorithm, we need to calculate the conditional mean of the data for each direction (expectation step) and we calculate this ignoring the auxiliary variables and the regularizing term. However, in the maximization step of the algorithm, we solve (15) with full regularization. Thereafter, it sends back the results $\mathsf{Y}_{kf_i} + \rho\mathsf{J}_{kf_i}$ to the fusion center (K values).

(D2) At the fusion center, The block matrix Z is updated by solving (24) for K blocks separately. Thereafter, with the updated Z, $\mathsf{B}_{f_i}(\mathsf{Z})_k$ is sent back to the i-th agent (K values).

(D3) At agent i, Y_{kf_i} for K values are updated using (17) and if stopping criteria are not met, we go back to (D1) above.

The initialization for the C-ADMM algorithm is done as follows. First, the initial values for J_{kf_i} can be taken as a block matrix of 2×2 identity matrices (or for a warm start, we can take the solutions from the previous time slot). The initial values for both Z and Y_{kf_i} are taken as $\mathsf{0}$. Because of this, the solutions obtained for J_{kf_i} at the first C-ADMM iteration for steps (S1) and (D1) will have an unknown unitary ambiguity (Yatawatta 2012a). Therefore, only at the first iteration, the averaging step in (S2) and (D2) should be done after projecting each J_{kf_i} to the mean value calculated using the quotient manifold structure described in (Yatawatta 2012a). For the remaining iterations, because Z and Y_{kf_i} are not $\mathsf{0}$, the unitary ambiguity will be common (for each direction) and normal Euclidean averaging can be done in steps (S2) and (D2).

The selection of the regularization parameter ρ (> 0) is specific to each problem and more detail can be found in Boyd et al. (2011). We note here that it is possible to select different values of ρ for different directions when K directions are calibrated. For instance, for source clusters that are far away from the phase center, it might be true that there will not by any smooth variation of the errors with frequency along that direction. Therefore, for that specific direction, we can make ρ very small (so no smoothness is enforced). On the other hand, for source clusters at the phase center (also at the center of the beam), we can safely assume that the errors vary smoothly with frequency and use a high value for ρ (typically $\in [1, 10]$). In section 4, we provide simulations where we have varied the value of ρ and see how the performance change.

The convergence of distributed calibration is discussed in appendix B in detail. This boils down to having a sky model with finite, non-zero flux and data with finite values, but the true sky can have zero flux, and then the solutions will be zero. Convexity of the cost functions are also desired for C-ADMM to converge (Boyd et al. 2011) and for an interferometric data model, this generally is assumed to hold.

The amount of information that needs to be exchanged between the i-th agent and the fusion center is $K \times 2N \times 2$ (complex variables). In contrast, the amount of observed data used in calibration at the i-th agent is of the order $N(N-1)/2 \times 2 \times 2 \times T$ for T time samples with $N(N-1)/2$ baselines. Therefore, when working with P frequencies, the total amount of data that needs to be accessed is $T(N-1)/2 \times 4NP$ and for K directions, the total amount of information that needs to be exchanged in distributed calibration is $K \times 4NP$. Hence, when $K \ll T(N-1)/2$, the amount of information passed is much less in distributed calibration, regardless of the value of P.

The total number of computations in distributed calibration compared to normal calibration is not significantly different, and in fact it could even be higher. However, we gain a significant reduction in operational and energy cost by being able to distribute the total computations across a network of compute agents. In addition, there are several possibilities to reduce the computational cost even further and these will be explored in future work. First, it is possible to eliminate the need for having a fusion center (Erseghe 2012; Shi et al. 2014) and design an algorithm where agents only pass data between their neighbours. Secondly, when there are data with more frequencies than the number of compute agents, a multiplexing scheme where each agent

alternates the data used in calibration, and yet calibrates the full dataset can be investigated.

4 SIMULATIONS

In this section, we present several simulations to illustrate the performance of distributed calibration. We consider a radio telescope similar to LOFAR, observing in the frequency range 115 MHz to 185 MHz with $N = 47$ stations pointed at the north celestial pole (Yatawatta et al. 2013). We consider data taken at $P = 32$ different channels (with bandwidth 0.2 MHz each) uniformly spaced in frequency within the observing frequency range. Therefore the frequency range covered by the data is 70 MHz wide but the actual bandwidth is 6.4 MHz. In a typical situation, in order to increase the number of constraints, calibration is performed for about every few minutes of data, using more than 1 time sample (for instance with 10 s integration, we have 30 samples for 5 minutes of data). In our simulations we only use 20 time samples in all calibration tests, equivalent to a total integration time of 200 s. Note that we call calibration of individual channels separately (without using the information across frequency) as *normal* calibration throughout this section.

We simulate (1) and the Jones matrices $\mathsf{J}(p,k,t,f)$ are simulated as follows. The variation with t is simulated as $\sin(\alpha_1 t' + 2\pi\beta_1) + \jmath\sin(\alpha_2 t' + 2\pi\beta_2)$ where $\alpha_1, \alpha_2, \beta_1, \beta_2$ are drawn from a uniform distribution $\mathcal{U}(0,1)$ for each station p and direction k, and t' is time sample number. The variation across frequency is simulated by using a polynomial $\sum_{l=1}^{G}(\gamma_l + \jmath\delta_l)\left(\frac{f-f_0}{f_0}\right)^{(l-1)}$ where γ_l, δ_l are also drawn from a uniform distribution $\mathcal{U}(0,1)$. The reference frequency f_0 is taken to be 150 MHz and $G = 4$ for all simulations. The product of the time variation and frequency variation gives the complete description of the elements in $\mathsf{J}(p,k,t,f)$.

The sky model consists of point sources, randomly distributed over a field of view of 7×7 square degrees. Their intensities at frequency f_0 is simulated using a power law and their spectral indices are drawn from a uniform distribution $\mathcal{U}(-1,1)$. The flux of the weakest source calibrated is set to be 1 Jy. The number of sources K simulated (and calibrated) is varied for different simulations as described below. In addition, we also simulate a set of 300 weak background sources, with peak flux below 0.1 Jy and have a flat spectrum. We do not corrupt these sources with errors because we want to examine the effect of calibration of the bright foreground sources on them.

Finally, we add noise N_{pq} to the simulated visibilities in (1). The elements of N_{pq} are drawn from a Gaussian distribution with zero mean and equal variance in real and imaginary parts. The noise variance is adjusted such that the total noise power is 10% of total signal power for the full observation, which is 6 hours.

4.1 Simulation I

We consider calibration along one direction $K = 1$. For normal calibration, we use 30 iterations of the RTR algorithm (beyond which we do not see any improvement). For distributed calibration, we use 50 C-ADMM iterations. Each C-ADMM iteration performs steps (S1,S2,S3) described in section 3 and step (S1) uses 10 iterations of the RTR algorithm. Let the simulated Jones matrices (4) at frequency f be given by J_f and its estimated value be given by $\widehat{\mathsf{J}}_f$. Then the error between J_f and $\widehat{\mathsf{J}}_f$ (per parameter) is found as $\frac{1}{\sqrt{4N}}\|\mathsf{J}_f - \widehat{\mathsf{J}}_f\mathsf{U}\|$ where U ($\in \mathbb{C}^{2\times 2}$) is a unitary matrix denoting the unitary ambiguity between the true parameters and estimated parameters. It is found by solving a matrix Procrustes problem (Yatawatta 2012a). We average the error calculated this way over the P frequencies and all time samples to get the final error.

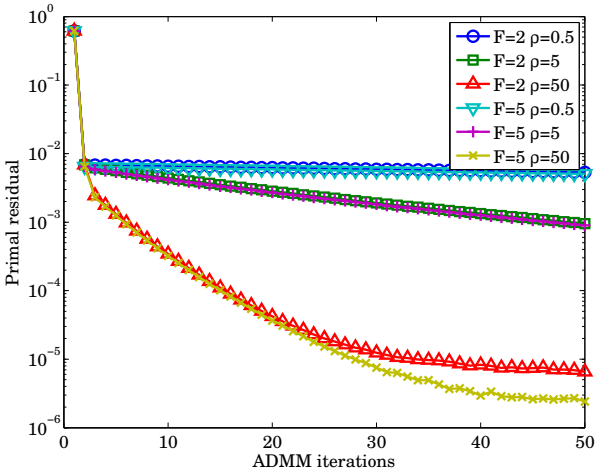


Figure 2. Variation of the primal residual with C-ADMM iteration number, for two values of smoothing polynomial terms $F = 2$ and $F = 5$ and three values of regularization factor $\rho = 0.5$, $\rho = 5$ and $\rho = 50$.

Moreover, we use two measures of error to study the convergence of distributed calibration. We define the 'primal' residual as $\frac{1}{\sqrt{4N}}\|(\mathsf{J}_{f_i})^n - \mathsf{B}_{f_i}(\mathsf{Z})^n\|$, averaged over all f_i. The 'dual' residual is defined as $\frac{1}{\sqrt{4PN}}\|(\mathsf{Z})^{n+1} - (\mathsf{Z})^n\|$, where the superscripts $n+1$ and n denote the C-ADMM iteration number. The primal residual depicts the error between the local solution and the predicted consensus value. On the other hand, the dual residual depicts the convergence of the global variable Z.

In Fig. 2, we have shown the variation of the primal residual and in Fig. 3, the variation of the dual residual, both with the C-ADMM iteration number. The regularization parameter is set at $\rho = 0.5$, $\rho = 5$ and $\rho = 50$. The number of terms in the smoothing polynomial (10) is set at $F = 2$ and $F = 5$, with $F = 2$ underestimating the simulated polynomial order while $F = 5$ overestimating it. It is clear that as the value of ρ increases, the primal residual converges faster, and to a lower value. Also, the dual residual is lower for a low order polynomial, or a lower value of F.

In Fig. 4, we show the average error per parameter for the chosen values of F and ρ, after 50 C-ADMM iterations. In all cases, distributed calibration gives a lower error than normal calibration. Even though the true parameters are simulated using a polynomial with $G = 4$, we get the lowest error for both $F = 2$ and $F = 5$, at $\rho = 50$. The lower bound of this error is determined by the noise and the weak sources not included in calibration. For this example, this bound is too high to see a difference in performance between $F = 2$

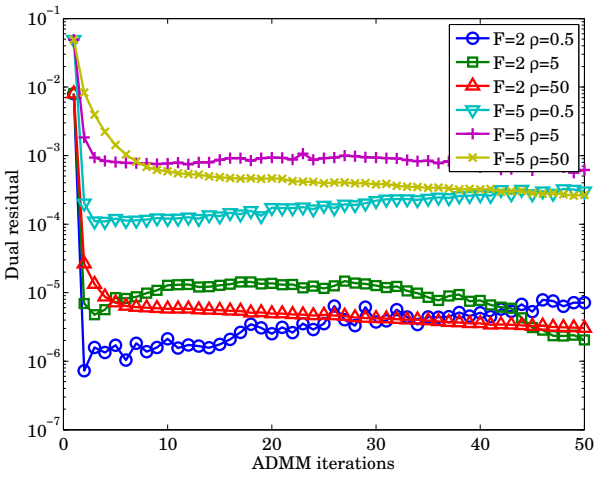


Figure 3. Variation of the dual residual with C-ADMM iteration number, for two values of smoothing polynomial terms $F = 2$ and $F = 5$ and three values of regularization factor $\rho = 0.5$, $\rho = 5$ and $\rho = 50$.

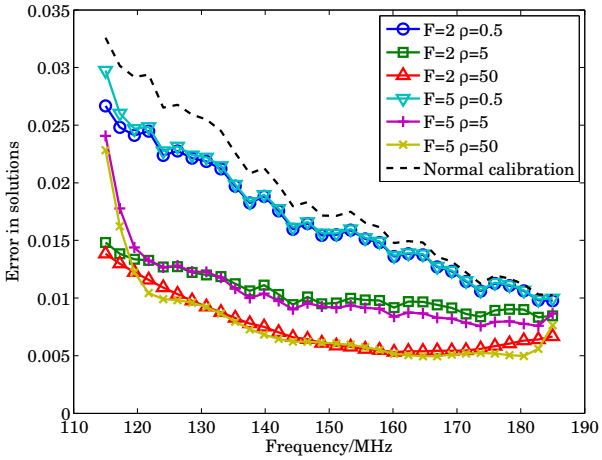


Figure 4. Variation of average error standard deviation of the estimated solutions with frequency after 50 C-ADMM iterations. Normal calibration has higher error and distributed calibration with $\rho = 50$ gives the lowest error, both for $F = 2$ and $F = 5$. The edge frequencies have higher error for $F = 5$ due to our choice of the interpolating polynomial.

and $F = 5$. Moreover, we also have errors due to polynomial interpolation, which is clearly seen for $F = 5$ at the edge frequencies.

4.2 Simulation II

In this simulation we set $K = 25$ and we use 20 time samples in calibration, in other words, calibration is performed for every 200 s of data. Therefore, for a 6 hour observation, calibration is performed 108 times. We use $F = 2$ and $\rho = 5$ and each calibration uses 20 C-ADMM iterations. In each C-ADMM iteration, there are 3 SAGE iterations. In Fig. 5 (a) we show the uncalibrated continuum image which is dominated by the errors along strong sources. In Figs. 5 (b) and 5 (c) we show the calibrated image after normal calibration and distributed calibration, respectively. The noise (at the edge) in Figs. 5 (a), (b), and (c) are 3.3 mJy, 0.64 mJy and 0.49 mJy, respectively. Therefore, there is a clear reduction in noise with distributed calibration, although this is not visible in Fig. 5.

In order to clearly show the difference, we have shown a small area of the full image in Fig. 6 where we show an area surrounding a bright source. The uncalibrated image is shown in Fig. 6 (a) and images after normal and distributed calibration are shown in Fig. 6 (b) and Fig. 6 (c), respectively. It is clear that both normal and distributed calibration does well in removing the source, and making the weak sources clearly visible. However, in Fig. 6 (b), there still is an error pattern at the location of the bright source. The magnitude of this error pattern is far below the noise floor of a single channel. Therefore, it is impossible to eliminate this error by normal calibration. However, as seen in Fig. 6 (c), distributed calibration does a much better job in removing this error pattern. This also explains the reduction of noise in Fig. 5 (c). We see similar error patterns in real observations (Yatawatta et al. 2013), and with distributed calibration, the quality of images can certainly be improved.

5 CONCLUSIONS

We have proposed consensus optimization as a way of performing radio interferometric calibration in a distributed way. Distributed calibration enables us to improve the quality of calibration as well as to distribute the overall computational cost. The aspect we used for consensus is the smoothness of calibration parameters over frequency. However, a similar strategy can also be adopted to exploit spatial and temporal smoothness that can be explored in future work. We have given simulation results to confirm the feasibility of distributed calibration and also the expected improvement in performance, for instance by avoiding converging to local minima in the optimization. Future work would address better interpolation schemes that enforce consensus as well as multiplexing schemes when the number of frequency channels that needs to be calibrated is higher than the number of available compute agents. The source code for the algorithms described in this paper is available at http://sagecal.sf.net/.

ACKNOWLEDGMENTS

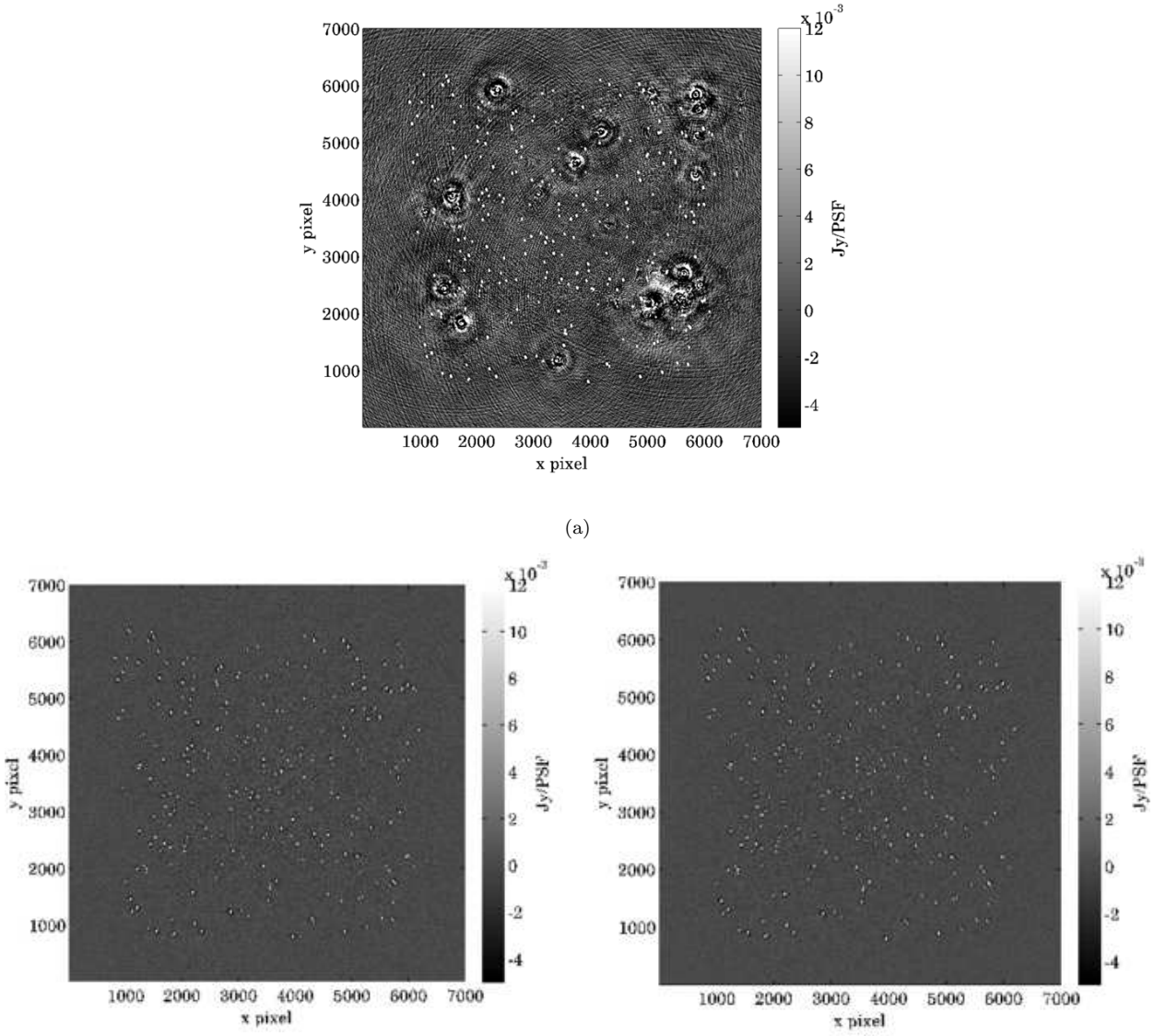
We thank the referee, Yves Wiaux, for the careful review and valuable comments. This work was financially supported by NWO (grant TOPGO 614.001.005) and the European Research Council (project LOFARCORE, grant # 339743).

REFERENCES

Absil P.-A., Baker C. G., Gallivan K. A., 2007, Found. Comput. Math., 7, 303

Absil P.-A., Mahony R., Sepulchre R., 2008, Optimization Algorithms on Matrix Manifolds. Princeton Univ. Press, Princeton NJ

Bertsekas D., Tsitsiklis J., 1997, Parallel and Distributed Computation: Numerical Methods, 2nd edition. Singapore: Athena Scientific



(a)

(b)

(c)

Figure 5. Continuum images of simulated data with $K = 25$ point sources with errors across a field of view of about 7×7 square degrees. (a) Raw image before calibration (b) Residual image after normal calibration (c) Residual image after distributed calibration. Both (b) and (c) are visually similar but (c) has lower noise.

Boyd S., Parikh N., Chu E., Peleato B., Eckstein J., 2011, Foundations and Trends® in Machine Learning, 3, 1

Bregman J., 2012, PhD Thesis, Univ. Groningen

Carrillo R. E., McEwen J. D., Wiaux Y., 2014, Monthly Notices of the Royal Astronomical Society, 439, 3591

Chang T.-H., Hong M., Wang X., 2014, Signal Processing, IEEE Transactions on, PP, 1

Erseghe T., 2012, Signal Processing Letters, IEEE, 19, 563

Ferrari A., Mary D., Flamary R., Richard C., 2014, in Sensor Array and Multichannel Signal Processing Workshop (SAM), 2014 IEEE 8th. pp 389–392

Fessler J., Hero A., 1994, IEEE Trans. on Sig. Proc., 42, 2664

Hamaker J. P., Bregman J. D., Sault R. J., 1996, Astronomy and Astrophysics Supp., 117, 96

Kazemi S., Yatawatta S., 2013, Monthly Notices of the Royal Astronomical Society, 435, 597

Kazemi S., Yatawatta S., Zaroubi S., Labropoluos P., de Bruyn A., Koopmans L., Noordam J., 2011, Monthly Notices of the Royal Astronomical Society, 414, 1656

Mota J., Xavier J., Aguiar P., Puschel M., 2013, Signal Processing, IEEE Transactions on, 61, 2718

Nocedal J., Wright S. J., 1999, Numerical Optimization. New York USA:Springer

Shi W., Ling Q., Yuan K., Wu G., Yin W., 2014, Signal Processing, IEEE Transactions on, 62, 1750

Tasse C., 2014, Astronomy and Astrophysics, 566, A127

Thompson A., Moran J., Swenson G., 2001, Interferometry and synthesis in radio astronomy (3rd ed.). Wiley Interscience, New York

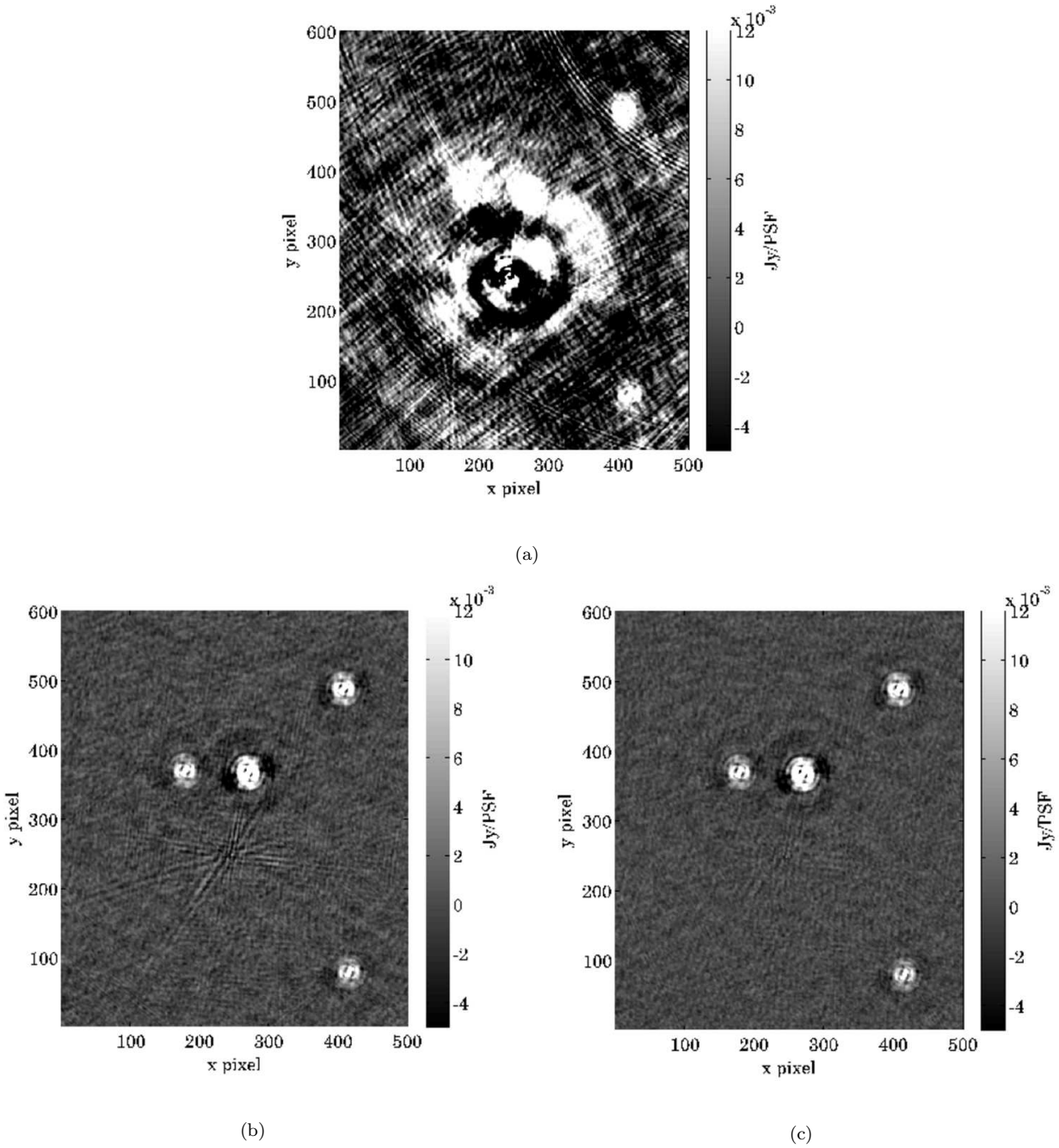


Figure 6. A small area of Fig. 5 showing a bright source. (a) Raw image before calibration (b) Residual image after normal calibration (c) Residual image after distributed calibration. In both (b) and (c) the errors due to the bright source have mostly disappeared, but (b) has a spike like error pattern that is also removed in (c).

Tsitsiklis J., 1984, PhD Thesis, MIT
Wei E., Ozdaglar A., 2012, in Decision and Control (CDC), 2012 IEEE 51st Annual Conference on. pp 5445–5450
Yatawatta S., 2012a, Monthly Notices of the Royal Astronomical Society, p. 33
Yatawatta S., 2012b, Experimental Astronomy, 34, 89
Yatawatta S., 2013, in Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. pp 3866–3870
Yatawatta S. et al., 2013, Astronomy & Astrophysics, 550, A136

This paper has been typeset from a TEX/ LATEX file prepared by the author.

APPENDIX A: GRADIENT AND HESSIAN CALCULATION

This section describes the derivation of the gradient and Hessian operators for (8) and (14) so that the Riemannian trust-region algorithm (Absil et al. 2007) can be applied. Without loss of generality, we drop the superscript f_i in this

section. We consider J to be on a matrix manifold denoted by $\mathcal{M}$. Let the function to be minimized be $g(\mathsf{J})$. We define the inner product for two elements in the tangent space $\mathcal{T}\mathcal{M}$ of this manifold as

$$h(\boldsymbol{\xi},\boldsymbol{\eta}) \triangleq \mathrm{trace}(\boldsymbol{\xi}^H\boldsymbol{\eta}+\boldsymbol{\eta}^H\boldsymbol{\xi}),\ \boldsymbol{\xi},\boldsymbol{\eta}\in\mathcal{T}\mathcal{M}. \tag{A1}$$

With this definition, the gradient is calculated to satisfy

$$h(\boldsymbol{\xi},\mathrm{grad}(g(\mathsf{J}))) = Dg(\mathsf{J})[\boldsymbol{\xi}],\ \forall\xi\in\mathcal{T}\mathcal{M} \tag{A2}$$

where

$$Dg(\mathsf{J})[\boldsymbol{\xi}] \triangleq \lim_{\tau\to 0}\frac{g(\mathsf{J}+\tau\boldsymbol{\xi})-g(\mathsf{J})}{\tau}. \tag{A3}$$

Similarly, the Hessian of $g(\mathsf{J})$ can be obtained as

$$\mathrm{Hess}\ g(\mathsf{J})[\boldsymbol{\eta}] \triangleq \lim_{\tau\to 0}\frac{1}{\tau}\left(\mathrm{grad}\ g(\mathsf{J}+\tau\boldsymbol{\eta})-\mathrm{grad}\ g(\mathsf{J})\right). \tag{A4}$$

Now we can rewrite (8) as

$$\begin{aligned} g(\mathsf{J}) = & \\ &\sum_{p,q}\mathrm{trace}\left(\left(\mathsf{V}_{pq}-\mathsf{A}_p\mathsf{J}\mathsf{C}_{pq}\left(\mathsf{A}_q\mathsf{J}\right)^H\right)^H\right. \\ &\left.\times\left(\mathsf{V}_{pq}-\mathsf{A}_p\mathsf{J}\mathsf{C}_{pq}\left(\mathsf{A}_q\mathsf{J}\right)^H\right)\right) \end{aligned} \tag{A5}$$

and we can rewrite (14) as

$$\begin{aligned} &L_i\left(\mathsf{J},\mathsf{Z},\mathsf{Y}\right) \\ &= g(\mathsf{J})+\frac{1}{2}\mathrm{trace}\left(\mathsf{Y}^H(\mathsf{J}-\mathsf{B}\mathsf{Z})+(\mathsf{J}-\mathsf{B}\mathsf{Z})^H\mathsf{Y}\right) \\ &+\frac{\rho}{2}\mathrm{trace}\left((\mathsf{J}-\mathsf{B}\mathsf{Z})^H(\mathsf{J}-\mathsf{B}\mathsf{Z})\right). \end{aligned} \tag{A6}$$

Finally, applying (A2) and (A4) to (A5) and (A6) gives us (18) and (19). Moreover, by taking gradient with respect to Z, we also get (22). Note also that since we use the RTR algorithm in Euclidean space, the projection is $\Pi(\mathsf{J})=\mathsf{J}$ and the retraction is $R(\mathsf{J},\boldsymbol{\eta})=\mathsf{J}+\boldsymbol{\eta}$.

APPENDIX B: CONVERGENCE

First, we consider the convergence of the RTR algorithm in minimizing a function such as $g(\mathsf{J})$ in (8) with respect to J. Using (Absil et al. 2008, 7.4.6), we only need to show that the manifold on which J lies (say $\mathcal{M}$) is compact (smoothness of $g(\mathsf{J})$ is obvious). Given that the sky model has finite and non-zero flux and the data has finite power, we see that $\|\mathsf{J}\|$ is finite and hence $\mathcal{M}$ is bounded.

Each pair of p,q in (8) gives us a set of constraints on the values of J that can be expressed as a set of nonlinear functions $\widetilde{g}_{pq,ij}(.,,\ldots.)=0$ for different values of p,q,i,j, which are actually mappings from $\mathbb{R}^{8N}$ to $\mathbb{R}$. Since 0 is a closed set, elements of J are in the inverse image of $\widetilde{g}_{pq,ij}(.,,\ldots.)=0$ which is also a closed set. Note that in order to have expressions such as $\widetilde{g}_{pq,ij}(.,,\ldots.)=0$ that are unique, we need to have at least few of them with nonzero values for C_{pq} and unique uv coordinates. In other words, we need to have a sky model with non zero total flux. For all possible values of p,q,i,j, with unique $\widetilde{g}_{pq,ij}(.,,\ldots.)=0$ expressions, we get an intersection of closed sets within which the elements in J must lie. Since the intersection of closed sets gives a closed set, we see that $\mathcal{M}$ on which J lies is also closed. Therefore $\mathcal{M}$ is both bounded and closed and by Heine-Borel theorem (Absil et al. 2008), it is compact.

For the convergence of the C-ADMM algorithm, we need to have smooth, convex functions for $g(\mathsf{J})$. This is not always guaranteed but with same assumptions as above, most of the time it can be safely assumed to be convex (see Yatawatta (2012b) for a detailed investigation).